

SEARCH RESULT SYNONYMY INDEXING FOR SOCIAL NETWORK USING LATENT SEMANTIC ANALYSIS

By

MOHAMAD FIRDAUS MOHAMAD ADIB

Dissertation submitted in partial fulfillment
of the requirements of the
Bachelor of Technology (Hons)
Information Communication & Technology

JANUARY 2014

Universiti Teknologi Petronas
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

SEARCH RESULT SYNONYMY INDEXING FOR SOCIAL NETWORK USING LATENT SEMANTIC ANALYSIS

By

Mohamad Firdaus Mohamad Adib

A project dissertation submitted to the
Information Communication & Technology Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Technology (Hons)
Information Communication & Technology.

Approved by:

Assoc. Prof. Dr. Mohd Fadzil Hassan
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

JANUARY 2014

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

Mohamad Firdaus Mohamad Adib

Acknowledgement

I bow my head before almighty Allah with gratitude. My indebtedness and salutation to many individuals who have helped shape this thesis are cannot adequately be conveyed in a few sentences.

First of all, I would like to express my gratitude to God for giving me the strength and health throughout my years of study in this university, that allows me to gain enough theoretical knowledge and practical skill sets as a technologist, to be applied into this particular project or other project in the coming years.

I would like to thank Assoc. Prof. Dr. Mohd Fadzil Hassan, my supervisor for this project for the past two semesters. Dr Fadzil Hassan has been giving me moral and mental support since the first day I proposed the idea of this project to him, in early semester September 2013. His understanding and confidence on my capabilities and passion towards the idea has given me the flexibility and spiritual advantages that I needed in conducting the project from start to finish.

Also, I would like to express my gratitude to my best friend, Fazri Mustafa, who helped me in giving me idea and introducing me to a new hope and passion, who have supported me throughout entire process, both by keeping me harmonious and helping me putting pieces together. I will be grateful forever for you as my dear companion.

There is always a sense of gratitude which one feels towards my mentor or guru who has helped me at one point of time or throughout. I shall be failing in my duties if I do not express my gratitude to other Faculty members and friends for their useful advice at various stages.

Abstract

Information retrieval (IR) has fashion the way of people acquiring information in Internet. Among of these are known as a search feature. Although semantic search is increasingly popular, not all the web has the technology to apply in their system mainly because of the various reasons of cost. Social network are known for their abstract and inconsistent of semantic data. This assessment is using term frequency vector of Wikipedia content to gather the most frequently terms within corpus as synonym data. After an overview of traditional search engine mechanism works and how synonym of a word associates meaning, the review solves broader and wider of data retrieval index by collecting same-meaning query in from search-data registry in social network by abstraction using Latent Semantic Analysis Synset. Conceptual relationships of set of query could be specified by taxonomy or it could be less passive inarticulate by statistical related to other words.

TABLE OF CONTENT

CHAPTER 1	4
1. INTRODUCTION.....	4
1.1. BACKGROUND STUDY	4
1.2. PROBLEM STATEMENT	7
1.3. OBJECTIVE OF ASSESSMENT.....	8
1.4. SCOPE OF ASSESSMENT	8
1.5. RELEVANCY OF ASSESSMENT	9
 CHAPTER 2.....	 11
2. LITERATURE REVIEW.....	11
2.1. SEMANTIC SEARCH AS THE FUTURE KNOWLEDGE DISCOVERY 11	
2.2. MECHANISM OF MODERN SEARCH ENGINE.....	12
2.3. LATENT SEMANTIC ANALYSIS IN KNOWLEDGE DISCOVERY	12
2.4. KNOWLEDGE DISCOVERY IN BIG DATA SCIENCE.....	13
 CHAPTER 3	 15
3. METHODOLOGY	16
3.1. RESEARCH METHODOLOGY.....	16
3.2. DEVELOPMENT METHODOLOGY.....	17
3.3. SYSTEM ARCHITECTURE DESIGN.....	22
3.4. DATA GATHERING AND ANALYSIS.....	22
3.5. EXPERIMENTATION AND MODELING.....	25
3.6. HUMAN COMPUTER INTERACTION AND USER INTERFACE.....	27
3.7. DATABASE SYSTEM.....	28
3.8. KEY MILESTONE.....	29
3.9. GANTT CHART	30
3.10. DEVELOPMENT TOOLS AND EQUIPMENT	30
 CHAPTER 4.....	 33
4. RESULT AND FINDINGS	33
4.1. DATA GATHERING AND ANALYSIS.....	33
4.2. COMPARISON TO TWITTER SEARCH FEATURE.....	35

4.3. SYSTEM TESTING	37
CHAPTER 5	38
5. CONCLUSION AND RECOMMENDATIONS	39
5.1. CONCLUSION	39
5.2. FUTURE WORK AND RECOMMENDATION	40
CHAPTER 6	44
6. REFERENCE.....	44
6.1. LITERATURES	44
6.2. DATASETS	45
CHAPTER 7	46
7 APPENDIX.....	46
7.1 DATA VALIDATION SURVEY.....	46
7.2 SOURCE CODES.....	50
7.2.1 <i>Wikipedia data miner and scraper.....</i>	<i>50</i>
7.2.2 <i>Tweets collector</i>	<i>52</i>

LIST OF TABLES

TABLE 1 : THE ASSESSMENT KEY MILESTONE	29
TABLE 2 : QUANTITATIVE SURVEY SCORE CARD	33

LIST OF FIGURES

FIGURE 1 : WORD MAP OF “BOOK”	6
FIGURE 2 : SCRUM METHODOLOGY PROCESS	19
FIGURE 3 : GINGER ARCHITECTURE BLUEPRINT	22
FIGURE 4 : WIKIPEDIA PAGE CONTENT SCRAPING I	23
FIGURE 5 : WIKIPEDIA PAGE CONTENT SCRAPING II	24
FIGURE 6 : TWEETS COLLECTION MIGRATION TO MONGODB.....	25
FIGURE 7 : THE ASSESSMENT MODEL AND COMPONENTS	26
FIGURE 8 : THE ASSESSMENT UI MOCKUP	28
FIGURE 9 : THE ASSESSMENT GANTT CHART	30
FIGURE 10 : QUANTITATIVE SURVEY RESULT	34
FIGURE 11 : TWITTER.COM SEARCH RESULT	35
FIGURE 12 : GINGER RESULT OF ‘JOURNAL’	36
FIGURE 13 : THE APPLICATION USER INTERFACE	37
FIGURE 14 : SYNSET MINING AND PROCESSING	41

CHAPTER 1

INTRODUCTION

1. Introduction

1.1. Background Study

Search is a lot about discoveries - the basic human need to learn and broaden the horizons. But searching still requires a lot of hard work by the user. The first web search engine, based on crawler was launched to public in mind 90s. Web-crawl has become the standard for all major search engines since and known widely to the public. It was the first version of search engines that only index web document from other websites by crawling through documents to other documents. Yahoo! was among earlier popular web search engine at that era and acted to be the largest directory for information seeking by browse the directory instead of doing a keyword-based search.

Later than that, modern search engine scene were roses to prominence when Google appeared in the search engine scene at 1996; The company keep changing the way search result indexing until achieved better results for many searches with an innovation called PageRank. Over a decade after that, concept of semantic search concept were introduced, to improving search accuracy by understanding user behavior and contextual to give user more relevant result. According to Readwrite.com, semantic search is search engine of the future. This modern concept mashed between natural language processing and machine learning would shapes the way human seeking for information. But anyhow, today, semantic search feature in major search engine provider are not wholly perfect yet. Lot of room of improvement can be done in semantic search to reach the most relevant of information query.

The growths of Internet users are increasing each year. Yet, the effectiveness in searching information was low. Users are unsure about the ways to achieve their goals. Only two ways to overcome this problem; the first one is education users to use product (search engine) right and another one is to build a right product. To educate people use technology right would bring more stages and require more commitment, but that should not what a technology do. Supposed, technology allows people to adapt to a new culture, not people have to adapt to embrace technology.

In embracing the technology to semantic web search, Latent Semantic Analysis (LSA) has taken into research on how to help people seeking information. LSA is a method that used in Natural Language Processing, a subset of Artificial Intelligence to approach linguistic in computer science, which intended to analyze relationship between set of terms in a document. LSA assumes that words in a document have a relation and similar meaning to the subject. In information retrieval (IR) application, occasionally it called Latent Semantic Indexing.

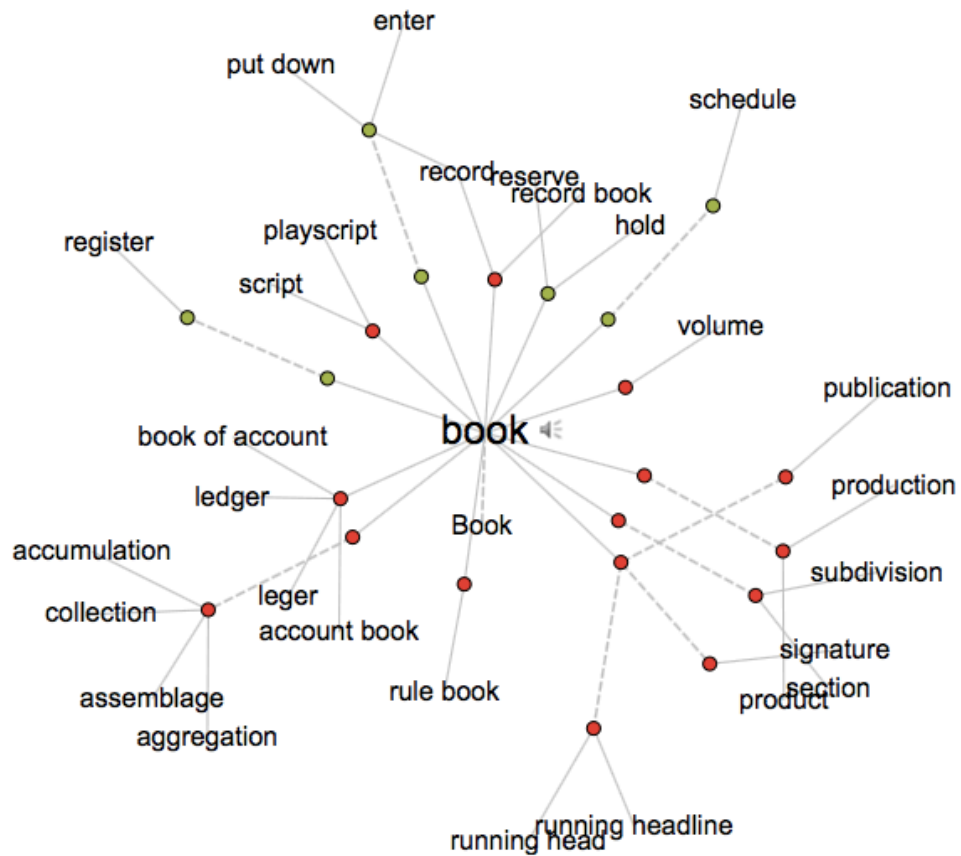


Figure 1 : Word map of “book”

Typically, to embrace technology to mankind and culture, the technology it has to be designed to deliver what people need. Semantic search allows users generally combine querying and browsing strategies to foster learning and investigation. Semantic search feature adapt various points including context of search, location, intent, variation of words, synonyms, generalized and specialized queries, concept matching and natural language queries to provide relevant search results.

1.2. Problem statement

Not every Internet users have good keyword selection in searching skills, especially inexperience and non-savvy Internet user. Internet users are ranging from all groups of ages and they are having different level of skill. In presence of social network like Facebook and Twitter, almost all of the information document-objects are undefinitive and abstract. Even modern search engine rich with feature unable to retrieve relevant information out of the pool because of ambiguities of information in social network.

Before semantic search were introduced, search engine works nifty by crawl or robot to entire city of web documents through pathways, called links. Once the engines found these pages and documents, next decode the details from them and store selected pieces in storages, to be recalled later when needed for a search query and return only those results that are relevant or useful to the searcher's query, and second, rank those results in order of perceived usefulness. It is both relevance and importance that the process of Search Engine Optimization is meant to influence.

Performing a search on social network is critical or users may end up with found nothing. The result returns are defined objective or phrase and undefinitive, as it only index exact or partial match based on the keyword in the query. User or searcher may found the result of they is not looking for. Results returned from a search sometimes are ambiguous to give user an answer to his query. Inconsistency is unacceptable in data transaction, users who used a search engine is expecting to receive a result within few second.

1.3. Objective of Assessment

The main objective of this assessment is to obtain a wider and broader results index in search mechanism and semantically sensible by terms to changes on how users interact with search results. By highlighting performance in result retrieval, Latent Semantic Analysis (LSA) is taking its place to deliver result index supported by elastic cloud computing. LSA is the method to mining synonym to accumulate a database which acted as databank for the query process regardless of the computing power. Therefore, it can be further into sub-objectives in order to achieve the main objective;

1. The project is to focus on design and architecture of the system and data mining preparation to the query process such as mining text from English Wikipedia.com and collecting social media data from Twitter.com.
2. To plan the successful of the system by returning synonym data based on keyword from the data. At this stage, this activity is to assess the availability of return synonym data.

1.4. Scope of Assessment

The area of study will be focus on the experience of user getting synonym document search result. This assessment is to prove early concept of semantic search, by return any document with same meaning through given keyword. At this stage, the activity is to conduct to answer how a document can be return semantically, regardless of relevancy. The activity involves the planning of two components; (i) user interface for user to query for information, (ii) which contain Meta info of synonym words known as synset. The heavy lift are at second component, consist of series of data mining from the large online encyclopedia, Wikipedia.com to analyze and collect synonyms as synset, and of rows of Twitter data as the domain.

Anticipating by this model would give a more relevant result to user compared with traditional database statement for typical SQL statement “SELECT * FROM table_name”. This approach, the set of query will be checked and return synonyms of the query entered against processed synset in the cloud database. The architecture of synset structure designed to enhance the performance of traditional search technologies, will return the linked data of a set of query before web interface response results using the synonym of the query.

The assessment domain is social network data, from micro-blogging platform Twitter.com, the focus is 140 characters text data, which is small enough to process the query.

1.5. Relevancy of Assessment

The project is relevant from different perspective as below:

1 End-user

Therefore, this study will help user to reduce burden to gather information easier through ambiguous data and enjoy the benefit of semantic feature in a search engine. The result will be indexed based on synonym meaning of keyword queried.

1 Developer

This assessment planned as a tools for developer to host this feature in their website or service. As planned, this feature will be packaged as a plugin to connect to the service through web interface Application Programming Interface.

2 Journalist (social media practitioner)

As practitioner who uses social media a lot, journalist will find this useful to find or swim deeper into social media data. This enable to find right information easily.

3 University and research society

Researcher believes that this assessment will direct and indirectly relevant to the objectives of all research institution. While semantic search is a new emerged topic discovered, researcher confident there is lot of room to be improved parallel to the time.

4 Researcher and computer scientist

As researcher is from computer science background, the study is relevant itself because the assessment is expected to produce a solution to seek information in ambitious data such as abstract information in social network. Besides that, research is expect to gain some degree of knowledge how semantic search can give broader result in information retrieval environment.

CHAPTER 2

LITERATURE REVIEW

2. Literature Review

2.1. Semantic search as the future knowledge discovery

For the past century, knowledge claims were vetted by professionals - those who grant degrees, edit journals, organize conferences, award grants, publish scholarly books, entertainment and in many other ways from casual to scientific progress of knowledge. Even a decade ago, students and scholar typically turned to scholarly journals and books in university library when they began their research project. The rise of search engine such as Google, has come in as the tool to find information they seek, and vast quantities of knowledge are available in one's own home or cyber cafe.

Information discovery has been evolved through the time. Search engines play an important role in controlling access to information and contribute significantly to the social construction of knowledge. The sociology of knowledge foundation improvised starting from the modern web search engine era and the technology has been changed the way human gathering information (Hinman, 2008). The new millennium of modern web search engine, semantic search has providing a better experience to end-users (Starr, 2013) whereas normally searching based on the natural language keyword as the how end-users speak to the search engine. Google is the one of the known search engine today made their move in providing semantic feature to their service. Paradoxically, the new technique of knowledge discovery has become new gatekeeper for the public general.

2.2. Mechanism of modern search engine

According to Wikipedia.com, web search function as a storage of information about many webs which retrieved from the HTML markup pages by a Web crawler sometimes known as a spider. This Web crawler will follows every link on the site exclude specific pages restricted by the site owner by using robots.txt. These pages will be analyzed by their contents to determine how it should be indexed, which then stored in an index database. Some web search store all or part of the source page, which referred as a cache while others store every word of every word they find. This cache would give benefits by keeping the original content of the page when the current page is updated or removed. The user can retrieved the pages that looking for by entering query. In response to that, the engine will examine its index and provide a listing of best matching web pages according to its criteria with a short summary containing document's title and a part of the web pages contained. Boolean operators such as AND, OR and NOT is applied to refine and extend the terms of the search. Apart from that, proximity search is introduced to enhance the search which allowing the user to define the distance between keywords. Another enhancement for the searching method to the user is concept-based searching where the research involves using statistical analysis on pages containing the words or phrases you search for. The usefulness of a search engine is determined by the relevance of the result search. Most engines employ ranking to the result in order to provide the best result first.

2.3. Latent Semantic Analysis in knowledge discovery

The early age of Latent Semantic Analysis (LSA) is focused on performance and scalability [11]. Relatively it requires high computational method and performance compared to another information retrieval (IR) method. The documents in concept indexing are represented using the popular vector space model. LSA reduces the overall noise in the semantic space accumulate words together using stemming and lemmatization. Hence after applying LSA some words share similar points in the semantic space, they are semantically similar.

The technique use term-document matrix, which count the occurrence of terms in document content as the formula below;

$$M_{tf-idf} = M_{train} \times M_{idf}$$

The term frequency occurrence are measured in this method, the score are result from the frequency of the terms divided by the total words count in the document (content). The assessment opted to use Wikipedia because it is currently the largest knowledge repository in the planet. In English version, the platform is available in 4.4 million articles (compared with 65k articles in Encyclopedia Britannica) [12].

2.4. Knowledge discovery in big data science

Semantic search looks to improve search accuracy by understanding searcher intent and the contextual meaning of terms as they appear in the searchable data-sphere, whether on the Web or within a closed system, to generate more relevant results. Rather than returning results based on matching keywords, the search engine is designed to match phrases based on it meaning, as well as objects on the site. Search results are based on both the matching set of query and associated with the query. This definitive feature is intended to promote users to have more broader and relevant results. By that, it can provide more information-rich object results and stimulate use of the feature.

In contrary of using ranking algorithms such as Google PageRank to predict relevancy, semantic search uses semantics, or the science of meaning in language, to produce highly relevant search results [8]. In most cases, the goal is to deliver the information queried by a user rather than have a user sort through a list of loosely related keyword results. However, the search giant Google itself has subsequently also announced its own Semantic Search project. Social media in general exhibit a rich variety of information sources. The search engine giant is not prior indexing a social content as a social media content are not optimized for search engine optimization (SEO). Engineers at

Facebook combining data-driven technique and scoring in their search to make it ranking more creative and semantically even more useful [5].

The assessment required dealing with substantial amount of data and processing resources to be processed. Google and major technology company has the technology and resource of semantic search, but not with small to medium size of sites. They did not have the same access and resources of what they can cater to their service like Google. Thus, this assessment will able to take dip into the same technology.

CHAPTER 3

METHODOLOGY

3. Methodology

3.1. Research Methodology

The research methodology requires gathering relevant data from the specified papers for literature reviews and gathering information from web search. These are the method and platform used to collect data and information about the project;

1. Technology, technical blogs and Internet

Association for Computing Machinery or ACM (<http://acm.org>) is the greatest and premier educational and scientific computing society, which provides and delivers resources that advance computing as a science and a profession. Also, through top technology engineering blog such as Facebook, Foursquare and Quora [9] give me large of exposure and clearer pictures of what today engineering challenge. From the industry trend and buzz, researcher recognized few issues and gap to be solved.

2. Academic research publishing

Through Google Scholar (<http://scholar.google.com>), researcher able to grab few researches and literatures on similar topic of what will assess. Also, through Facebook (<https://www.facebook.com/publications>) and Google publishing (<http://research.google.com/pubs/papers.html>) give more insight in current technology even some of them are unaffordable and uneconomic for undergraduate level.

3. Advice from experts

Researcher seeks experts review and advice, in both industry and academic. In industry, researcher received numerous of feedbacks from colleague and engineer during industrial training about the ideation and method what should be used for the execution. In academic, researcher seek theoretical and conceptual advice from supervisor, lectures and also postgraduate who working on the same niche; web semantic.

4. Empirical studies, personal experience and keen expertise in data science

As the university not caters much this areas (informatics and data science), these studies are conducted unsupervised. By took few courses about data science in Cousera (<http://coursera.com>) and read few titles on the particular topic helped researcher to have deeper understanding on the research area.

3.2. Development Methodology

There are two methodology used for this assessment, the first one is Extreme Programming, used for the development and Minimum Viable Product for the development product template. Both are focussing on agility in development and research.

Extreme Programming

The development approach that has been using in this practices is extreme programming (XP) methodology. XP is a subset of agile development methods which based on iterative and incremental development, where requirements and solutions evolve through phase

and development stories between self-organizing, cross-functional with product owner to improve quality of the application and responsiveness to the requirement changing. XP enables the creation of self-organizing development by encouraging verbal communication between stakeholder and disciplines in the project. XP manifesto actively pursuing communication, simplicity, feedback, courage, and respect.

In this development approach. Researcher took some element from scrum where break the large development into chunks called stories. Scrum defined to promotes adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, and encourages rapid and flexible response to change. The reason why I cannot apply scrum as a whole because this is a solo ride and scrum work best in collaboration across a team. Thus, a little of amendment had to do to suit the constraints - became “cowboy scrum” or in laymen define, scrum-for-one. However, the method will used is still agile development methodology, where using the element of iterative and incremental development approach.

Basically, the methodologies are involved;

1. Test-driven development
2. Developing in small sprints
3. Having a lot of contact and consult with the supervisors

Since the development will be solo and uncollaborative, developer have to seek constant feedback for every clear defined and short term goals from supervisor, who is also the product owner in this case. Even as a solo developer this comes in very handy because the sooner developer identify problems, whether it is requirements, design or implementation - the less expensive it will be to fix those problems. It shall called 'Cowboy Scrum'.

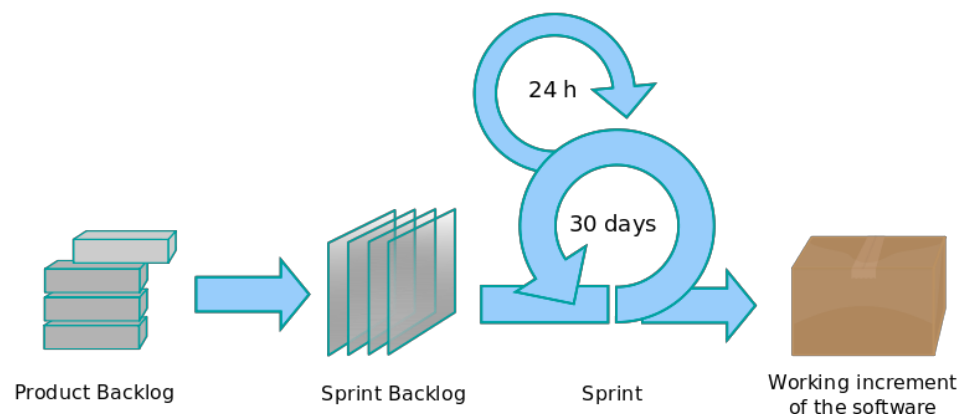


Figure 2 : Scrum methodology process

It as usual scrum, everything is almost the same except there is no greater number of single developer working on development. In this modified approach. There are two core roles in scrum-for-one methodology:

Product owner

The product owner is represents stakeholders and act as the voice or the customer and end user. A product owner is accountable for ensuring to drive the team to delivers value to the products cum to business. The product owner set the customer-centric items, typically called as user stories, ranks and prioritized them, before submit to the product backlog.

Most of the time, the team have to seek for clarification and definition with the product owner if stuck at some point or reiterate the cycle.

“Sprinter”

The sprinter is the one who accountable for setting and delivers the sprint objective and deliverables. Sprinter also known as the developer. A sprinter must ensure as intended to use rules of scrum, champion series of key meetings with product owner and challenge him to sprint on stories of development. Sprinter also responsible for deliverables of product increments for each sprint goal.

Minimum Viable Product

The method has been popularized by Eric Reis from his book, *The Lean Startup*[10]. It is a strategy to rapid prototyping and validating the market of a product or product feature. The idea is to have the barest minimum number feature to test the concept for semantic search using synonym. By adapting agile, the methodology is consisting of three blocks; idea, code and data.

- **Ideation**

The stage where the process of generating, building, and communicating business and technology ideas. For this assessment, based on the feedback from FYP I last semester, the idea was to use the most economical way collect synonym data as the synonym repository. Wikipedia has an excellent structure of crowdsourcing content and available in public, which in the sense to build a good synonym dataset based on Wikipedia articles. The problem is, Wikipedia English alone has over four million (44.6 Gigabytes) of

crowd sourced content and it will took lot of processing power and time to process them.

Another issue, a search engine, typically expected to return results in a split second, it is almost impossible to do in the real time if it has to mine the large of Wikipedia data, and the risk of being inconsistency. The best way and economic is to mining the Wikipedia documents separately and store as data repository where can be accessed by by the index engine for getting the results.

After processed the document from Wikipedia. A set of data will be collected in an array where it will stored as Javascript Object Notation so that web application (interface) can access the repository anytime.

- **Code (develop)**

The objective is to translate the idea into viable product, to proof the concept and mostly through by visualised it. The development involved series of feedback from many parties such as Final Year Project supervisor, experts from industries and academicians who have expertise in Artificial Intelligence and Data Mining.

- **Data**

To test the concept, we need document to be query by end-users. Here, social network data has taken into place. Twitter.com, famous micro-blogging platform, limited to 140 characters known as a tweet, limiting the clause to be indexed. To have a balanced search query, enough of tweets are needed to be collected. To test the concept, as the long text of tweets is limited, the chances to grab search content are also limited. Hence, enough domain data are needed.

3.3. System Architecture Design

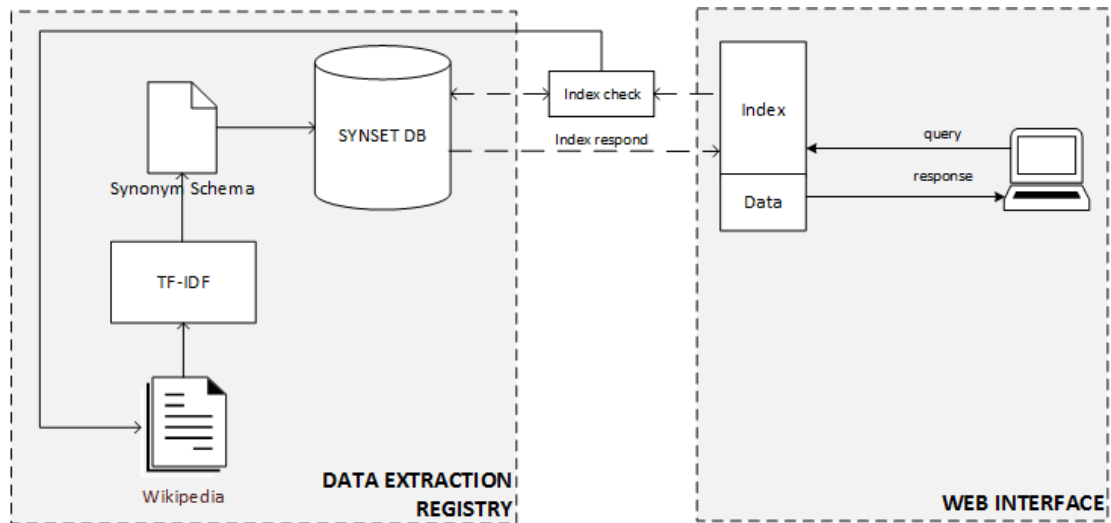


Figure 3 : Ginger architecture blueprint

This assessment is consisting of two components, data extraction/registry and web interface. The heavy lift part is at this first component, Data Extraction.

3.4. Data Gathering and Analysis

Even though data science is today buzzword, it is not a newcomer in computing and science field. However, it is new emerging knowledge, which consist of many disciplines. I put this project in my mind at the first place is to validate and accommodate what had I learn since past two years in data science. Search semantic is still huge and this project is the tip of icing.

The process behind the show is straightforward and unsophisticated. There are two separate components to communicate each other. The heart of component is the data extraction registry, where the component is located in cloud to take the advantage of an extensive database and elastic computing power; to mine and register synonym chain called synset from sets of rich-object document. The main objective of taking advantaged the power of cloud computing is because a social network are growing by pattern, and by having a good practice to compute and host specific computer to serve to the platform is a resource-effective [5].

1. Mining from Wikipedia

The execution is to write a Python script to scraping and mining content with single keyword from English Wikipedia.com.

The URL: *http://en.wikipedia.com/wiki/{keyword}*

The screenshot shows the Wikipedia page for the article "Software". The title "Software" is at the top, with a red arrow pointing to it and the word "keyword" written next to it. Below the title is a red box containing the main text of the article. The text describes computer software, its components, and its history. The text is as follows:

Computer software, or simply **software**, also known as **computer programs**, is the non-tangible component of **computers**. Computer software contrasts with **computer hardware**, which is the physical component of computers. Computer hardware and software require each other and neither can be realistically used without the other.

Computer software includes all computer programs regardless of their architecture; for example, **executable files**, **libraries** and **scripts** are computer software. Yet, it shares their mutual properties: software consists of clearly-defined instructions that upon execution, instructs hardware to perform the tasks for which it is designed. Software is stored in **computer memory** and cannot be touched, just as a 3D model shown in an illustration cannot be touched.^[1]

At the lowest level, executable code consists of machine language instructions specific to an individual **processor** – typically a **central processing unit** (CPU). A **machine language** consists of groups of binary values signifying processor instructions that change the state of the computer from its preceding state. For example, an instruction may change the value stored in a particular storage location inside the computer – an effect that is not directly observable to the user. An instruction may also (indirectly) cause something to appear on a display of the computer system – a state change which should be visible to the user. The processor carries out the instructions in the order they are provided, unless it is instructed to "jump" to a different instruction, or interrupted.

Software is usually written in **high-level programming languages** that are easier and more efficient for humans to use (closer to **natural language**) than machine language.^[2] High-level languages are compiled or interpreted into machine language object code. Software may also be written in a low-level **assembly language**, essentially, a vaguely mnemonic representation of a machine language using a natural language alphabet. Assembly language is converted into object code via an **assembler**.

The screenshot also shows the Wikipedia sidebar on the left with various links and the "Contents" table of contents on the right.

Figure 4 : Wikipedia page content scraping I

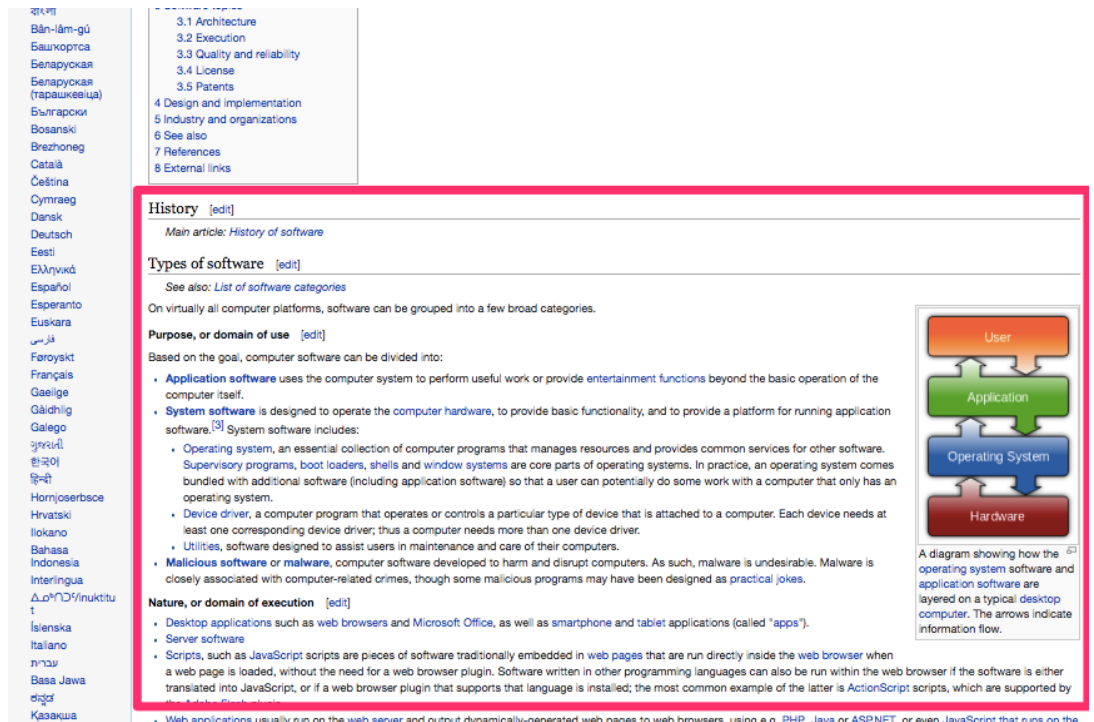


Figure 5 : Wikipedia page content scraping II

Next, the script will mining and process by count the frequency of top 15 of the most common (term frequency) used words and save it in a JSON files. The text is only the content of the keyword (refer the red box above). Skips else than content such as header, footer, menu, sidebars and etc.

2. Tweets collection

The tweets are mined through open source script by gdelfresno, which can be fork out through his Github

(<https://github.com/gdelfresno/twitterstream-to-mongodb>).

The process is straightforward, you need an access to use their API and simple setup.

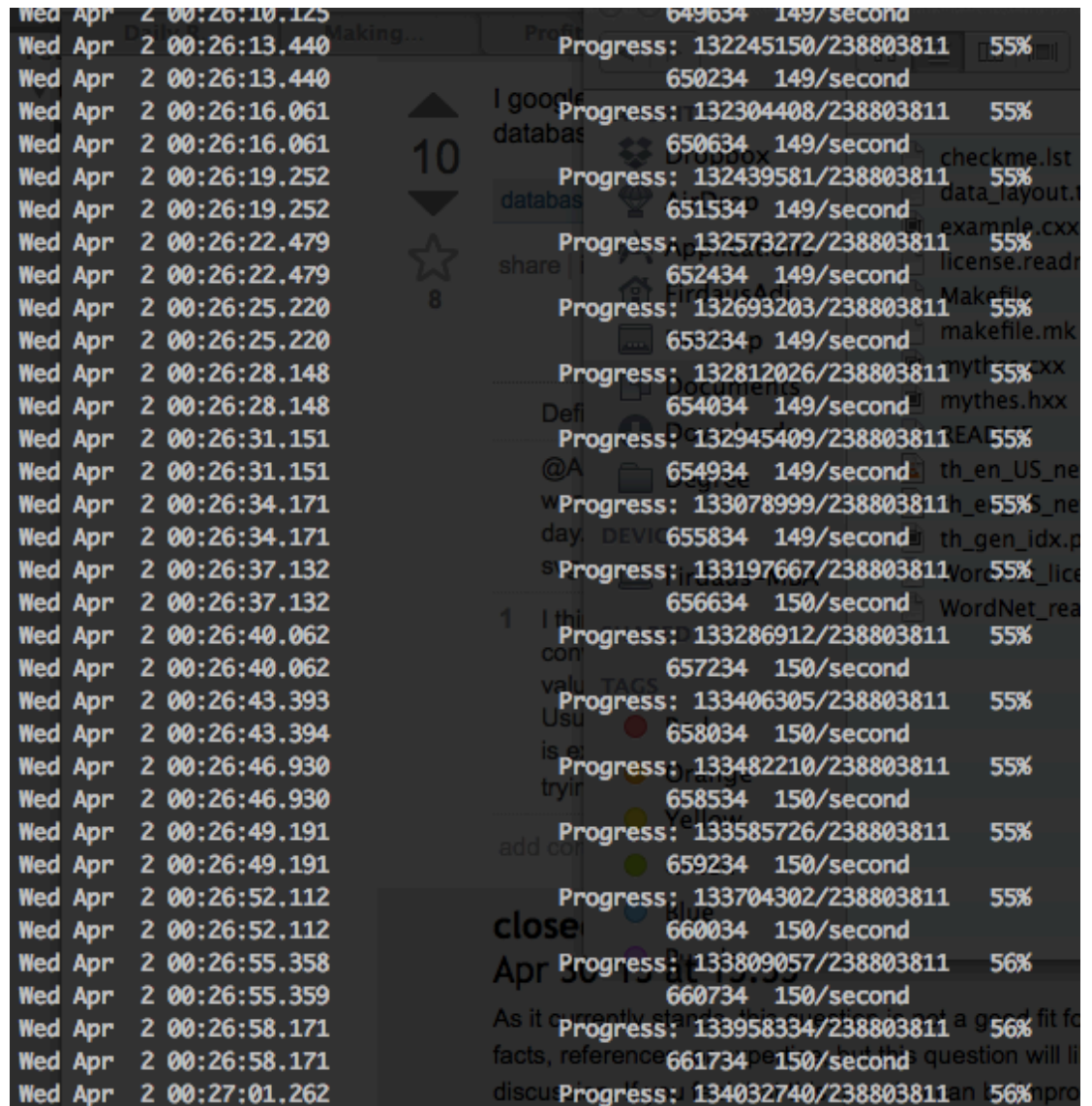


Figure 6 : Tweets collection migration to MongoDB

3.5. Experimentation and Modeling

At this point, two major component and data mining are done. The development plan will be progressively increment according stories as initiated earlier. The logic and integration are laid out as follow:

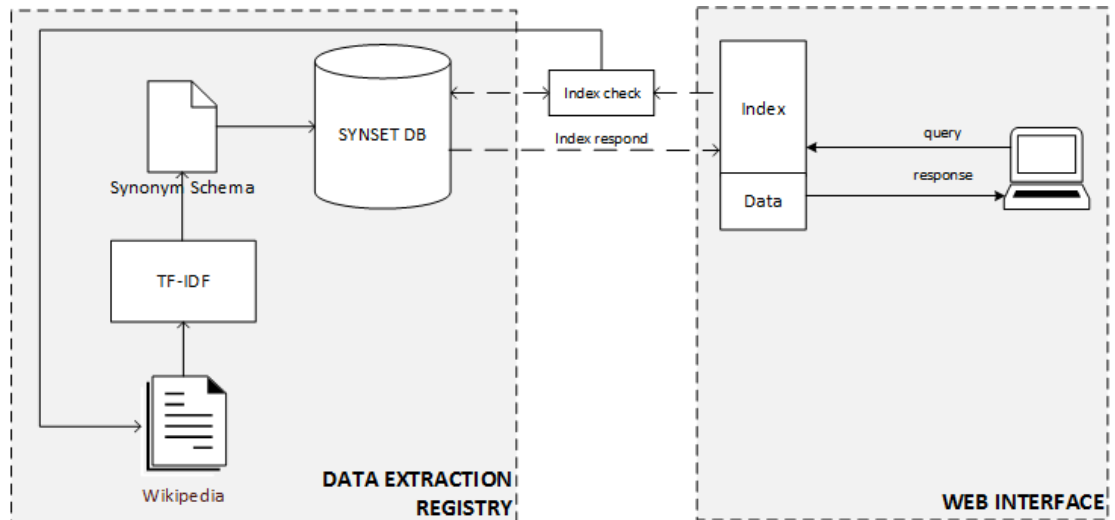


Figure 7 : The Assessment models and components

The first component, Data extraction and registry is where the heavy lifting is. This component is focusing on the synonym engine where the synonym repository is stored. First, a script will mined and processed English Wikipedia pages by collecting the most frequent terms in the documents, after that the terms will be stored as synonym schemas known as JavaScript Object Notation file. Later, it will be imported into Synset database where the synonym repository is. This repository will be access by index checker of web application to get an array of synonym file. The logic behind the most frequent terms of Wikipedia document is typical relationship of the content is associated with the document. This is the part where Latent Semantic Analysis taken part, by assumes that words that are associates in meaning will frequent repeated in similar pieces of text.

The second component is the end-user's User Interface. Here is the where end-users interact and make query to request information. The communication between of two component is by API by translating synonym of the query and return an array contained synonym of the words before drill down for results.

This assessment is not meant for relevancy over precision or accuracy but completeness. Anticipating by this model would give a more relevant result to user compared with traditional database statement for example SQL statement “*SELECT * FROM table_name*”. In this approach, the set of query will be

checked and return synonyms of the query entered against processed synset in the cloud database. Namely high computation primarily designed to enhance the performance of traditional search technologies, will return the linked data of a set of query before web interface response results using the synonym of the query.

3.6. Human Computer Interaction and User Interface

During planning phase, researcher trying to keep the user interface as minimum as possible. The mockup showed the tweets page only list of tweets and a search bar where user can enter query of their choice to seek information.

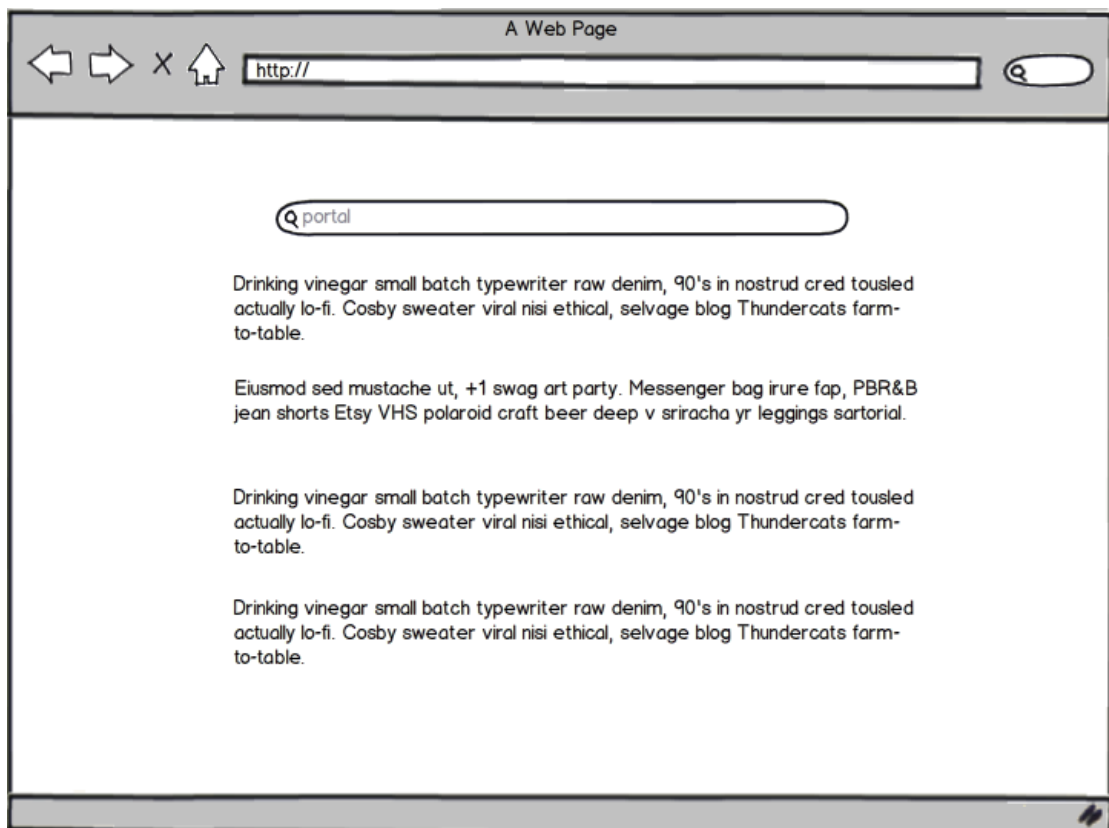


Figure 8 : The assessment UI mockup

Researcher tried to inherit the famous web search giant, Google.com by their minimalistic of their product.

3.7. Database System

Two data inventory involve in this assessment; tweets and synset (synonym repository). Both of them are stored in NoSQL and JavaScript Object Notation.

Tweets data

This is the data object structure of the tweet data, followed by the model schema :

```
{
  "_id": {
    "$oid": "533ad62a68ccb179b14d3ac2"
  },
  "key": 4,
  "rank": 4,
  "time": "Mon May 11 03:18:03 UTC 2009",
  "hashtag": "kindle2",
  "handler": "vcu451",
  "tweetdata": "Reading my kindle2... Love it... Lee child's is good read."
}
```

Twitter	
Model: Tweets	
Id	:string
Key	:int
Rank	:int
Time	:string
Hashtag	:string
handler	:string
tweetdata	:string

Synset data

Synset data are collected into JavaScript Object Notation (JSON) at first, before port into MongoDB for storage and accessibility from the application. This are the sample of JSON object, followed by the model schema of the database:

```
{ "wok": [ "wok", "stove", "cooking", "chinese", "food", "heat", "used", "steel", "iron", "cast",  
"also", "frying", "traditional", "handle", "stir" ] }
```

tf	
Model: Synonym	
Synonyms	: Array

3.8. Key Milestone

Details	Progress Period
Data mining and dataset setup	Week 2-6
Progress report	Week 4
Logic and backend development	Week 7-10
Frontend development, integration and testing	Week 11-13
Pre-SEDEX	Week 11
Dissertation	Week 12-14
Submission of Technical report and dissertation	Week 13
VIVA FYP	Week 15
Hard bound submission	Week 16

Table 1 : The assessment key milestone

3.9. Gantt Chart

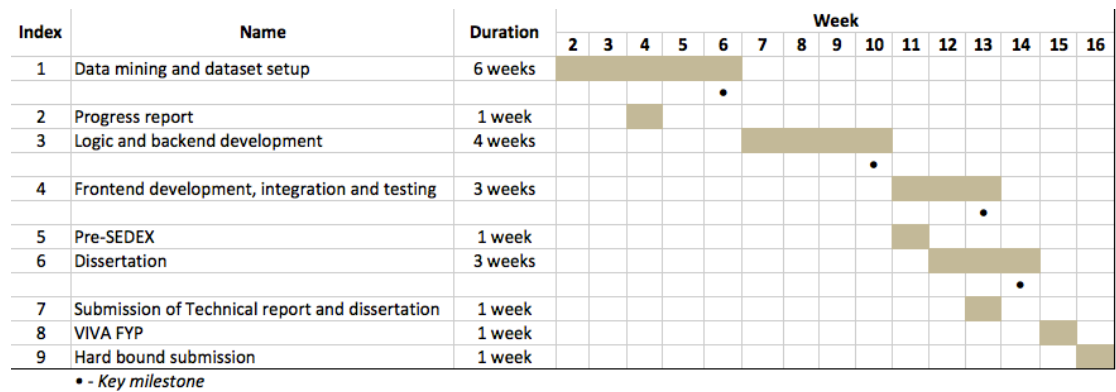


Figure 9 : The assessment Gantt chart

3.10. Development Tools and Equipment

Hardware

- **Macbook Air 11 Mid 2013** for the primary machine to console process in cloud. It is portable and most of the time used for development.
- **Macbook Pro 15 Early 2012** for the data cleaning and processing. This is also where data were downloaded.

Hosting

- **Digital Ocean** (<http://digitalocean.com>) for the SSD cloud based infrastructure.
 - Run on Ubuntu 12.04.3 x64
 - 512MB Ram
 - 30GB SSD Disk
 - Data center is located at Singapore, which is closest to to access.

- **Heroku** (<http://heroku.com>) for Ruby on Rails host server to host the app.

Software

- **Github** is the code repository for web-based hosting service for software development projects that use the Git revision control system.
- **Sublime Text 3** is a sophisticated text editor for code, markup and prose. This is where the development and code review part were handled.
- **iTerm** is a Mac OSX terminal which is focus on performance, internationalization, and supporting customised features to enable productivity.
- **Evernote** is note taking application for literature and ideas collection.
- **MongoDB** is an open-source schema-less document database for NoSQL. The database platform is used to support Synset repository and Tweets data.
- **Homebrew** (<http://brew.sh>) is package manager to install packages and development component for OSX.
- **Asana** is used to collaborate, getting feedback and as agile project management tool during assessment and development.
- **Balsamiq Mockup** for rapid prototyping and visualization tool. The tool is use to sketch and design the interface before implement to code.

Programming Language

- **Ruby** and **Ruby on Rails** for the web backend and interface.
- **Python** for data analysis tools to process and handle numerical as well as big data. For this assessment, few well known on-shelf Python packages were used such as Numpy, NLTK (for handling Natural Language Processing) and Scrappy (to handle data capture from web). Python also known as a scientific programming language for scientist.
- **SASS** and **AJAX** for the front-end development and cosmetic to the user interface.

CHAPTER 4

RESULT AND FINDINGS

4. Result and Findings

4.1. Data gathering and analysis

This assessment is called Ginger, the code name for the research title of Search Result Synonymy Indexing for Social Network Using Latent Semantic Analysis.

As the method used, Latent Semantic Analysis – a short survey has been conducted to verify the sets of collected terms are associates with the keywords. The quantitative survey is using empirical method which consisting of ten respondents, in a set of ten random synset schema each respondent. This method which assumes cluster of the 100 thousands of collected synset schema by taking 1 percent of total clause to get feedback has collect. In this survey, we get feedback of random 100 synset schemas to represent of whole clause of collected synset. The sample of question is shown in **Appendix 7.1**.

Respondents	Synset schema scores										Average
	A	B	C	D	E	F	G	H	I	J	
Respondent 1	6	7	9	10	6	5	6	7	8	10	7.4
Respondent 2	10	10	8	6	8	5	4	4	9	8	7.2
Respondent 3	6	8	9	8	9	9	7	7	7	5	7.5
Respondent 4	9	8	7	7	8	6	7	6	10	8	7.6
Respondent 5	10	3	4	5	7	9	8	7	8	5	6.6
Respondent 6	6	5	4	7	8	10	9	8	9	9	7.5
Respondent 7	8	5	4	4	9	8	7	7	8	6	6.6
Respondent 8	9	8	9	9	7	6	7	9	10	6	8
Respondent 9	7	7	8	6	9	10	10	8	6	8	7.9
Respondent 10	2	7	6	8	7	5	8	8	7	9	6.7

Table 2 : Quantitative survey score card

The average score of all synset schemas is 7.3. The range of the average score per respondent is 60 to 80 percent of the synonym collected is accurate, which is high using Latent Semantic Analysis method. It is subjective and depending on the documents and dataset, Using Latent Semantic Analysis method, the scores weight can be considered high (Magerman, Looy and Song, 2008).

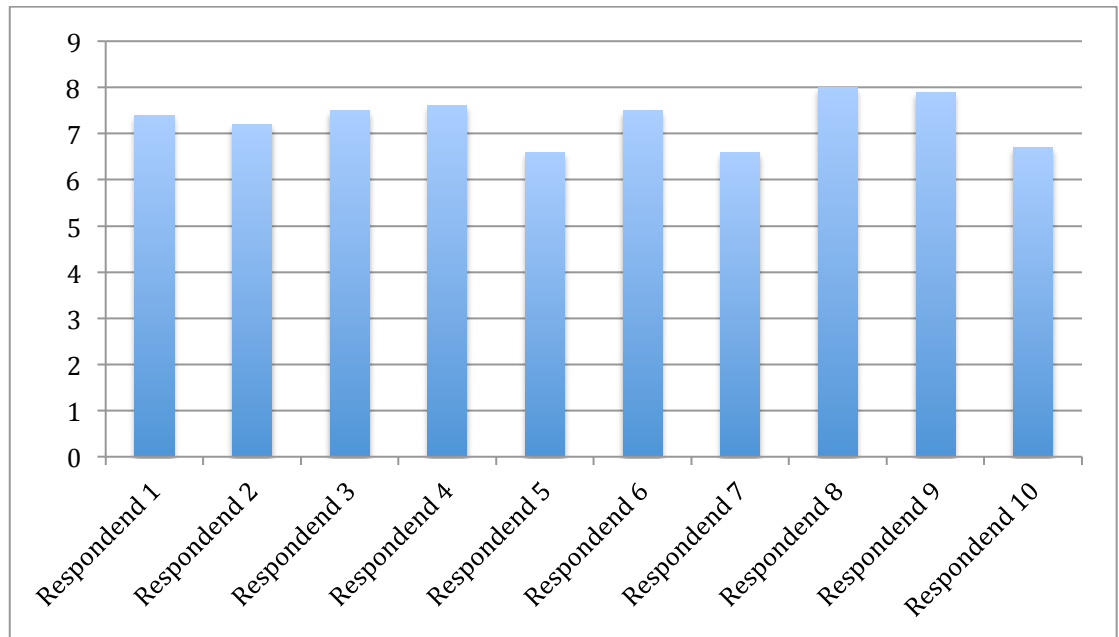


Figure 10 : Quantitative survey result

Nonetheless, the relevancy of the collected data is depending on documents that been processed. Latent Semantic Analysis is used to count terms frequency. Through the result, the objective is answered by completeness. Even though the assessment are focusing on completeness, not on relevancy of the result. An attempt to minimize the precision threshold in the research. As the bigger the range of the most term frequent, the precision are gone loose and irrelevant.

4.2. Comparison to Twitter Search feature

The closest comparison to look is at the social network itself. Twitter has their search feature called Twitter Search. Below is the result from Twitter.com of the query “Journal”.

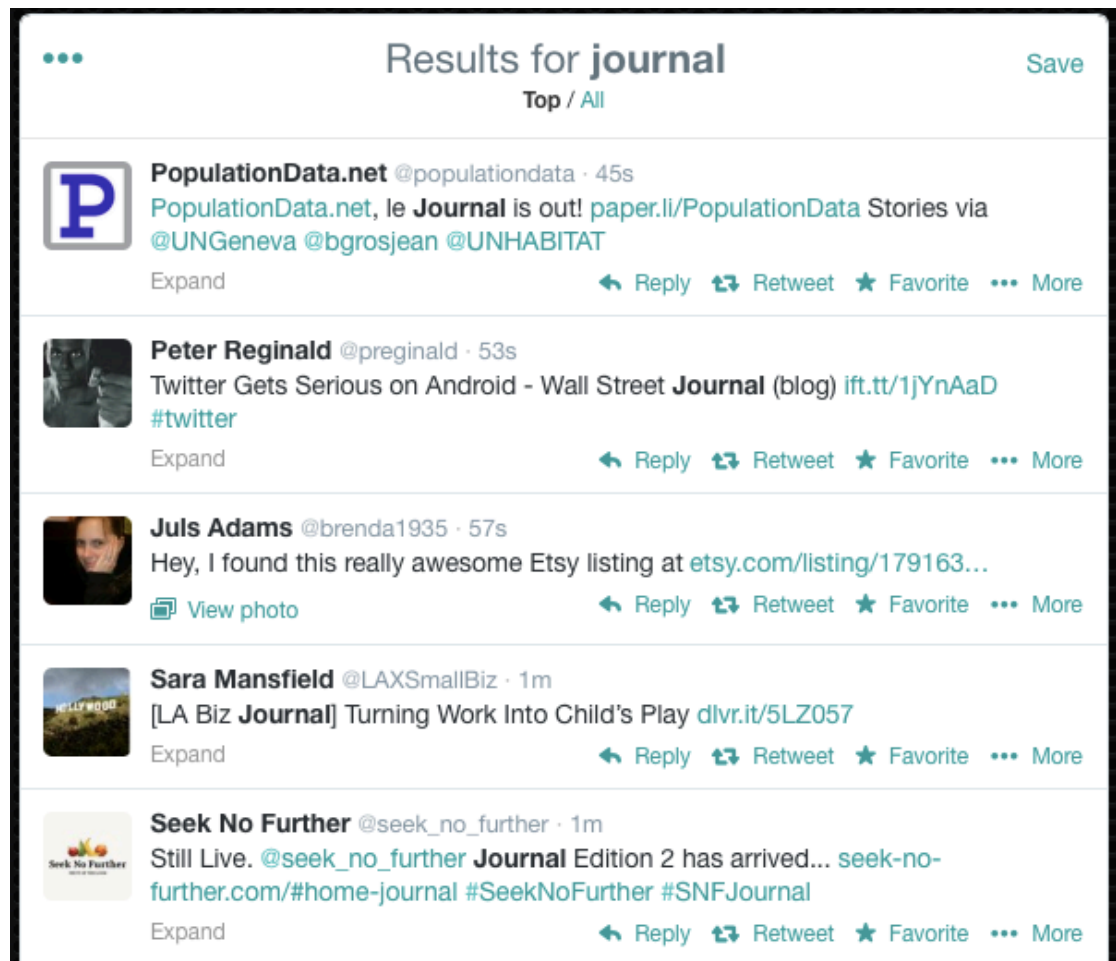


Figure 11 : Twitter.com search result

The URI of <https://twitter.com/search?q=journal> returned tweets containing word ‘Journal’ and it does same ways on how traditional search engine like Google works.

Take a closer look at semantic search engine of what return from the synset schema of the same keyword:

```
{"Journal": ["journal", "public", "business", "record", "also",  
"magazine", "transaction", "daily", "writing", "day", "see", "event",  
"called", "scholarly", "reference"]}
```


The search results from Ginger, where it returns result using word association from the array of synset, index tweets from the array of synonym words.

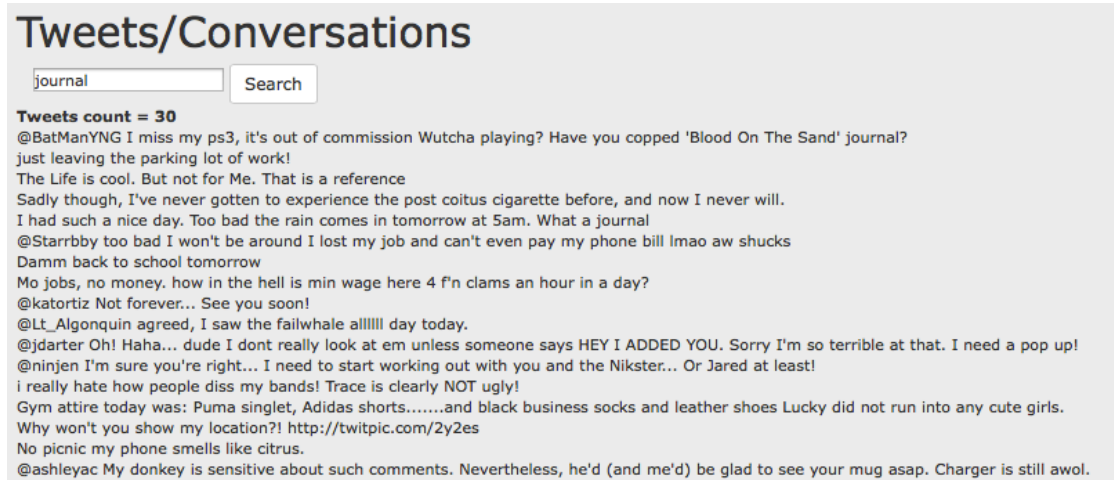


Figure 12 : Ginger result of 'Journal'

4.3. System Testing

The system is deployed to Ruby on Rails cloud server, called Heroku (<http://heroku.com>). The application can be accessed at <http://gingerclover.herokuapp.com>.

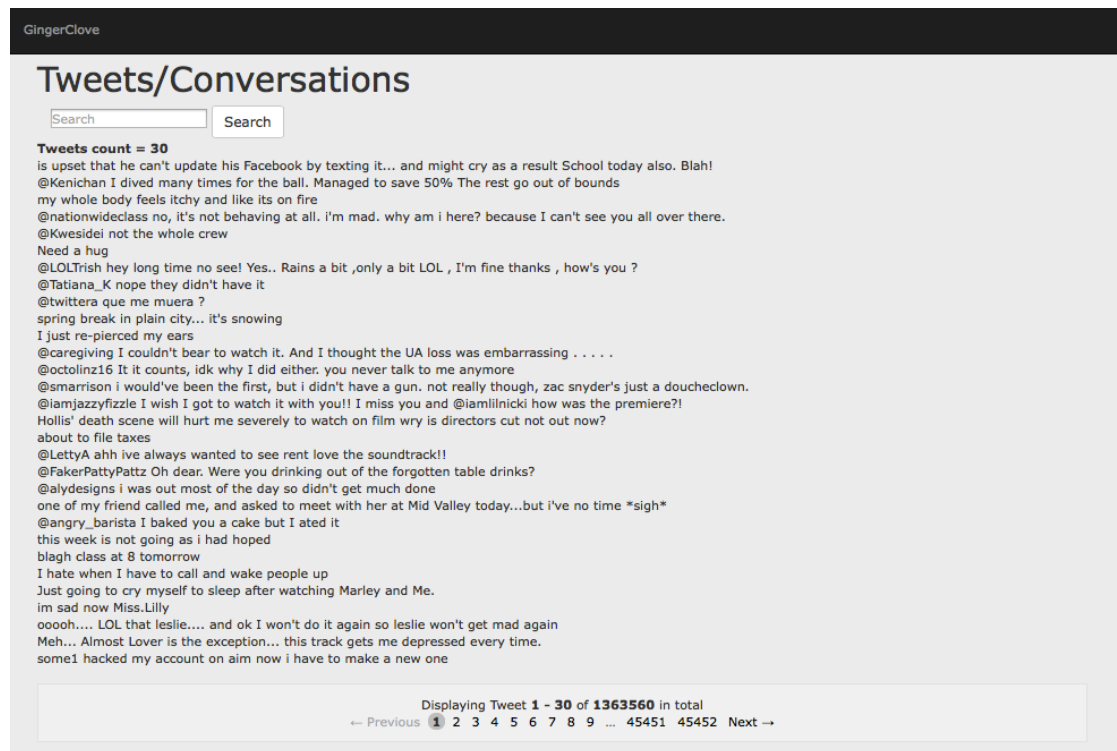


Figure 13 : The application user interface

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5. Conclusion and Recommendations

5.1. Conclusion

The basic structure of this assessment is consisting of two parts; first component is data mining and extraction from sets of internet document object (<http://wikipedia.com>) into schemaless database to register and check the index data and return it to the web interface to query up the results. Another component is the web interface for users to key in the set of query before checked against the synset database. This is where the normal search-return index works.

To prove this assessment, development will be taking place to answer the concept. The idea of the search mechanism will return a more meaningful and broader results the information retrieval and exploratory domain. During the assessment, most time are consumed on the literature and the first component, data mining. The data mining consist of two sets, the first is on preparing synonym schemas using Latent Semantic Analysis on Wikipedia documents and the second part is to mining micro-blogging data from Twitter.com to be used as the subject and domain of the assessment and research.

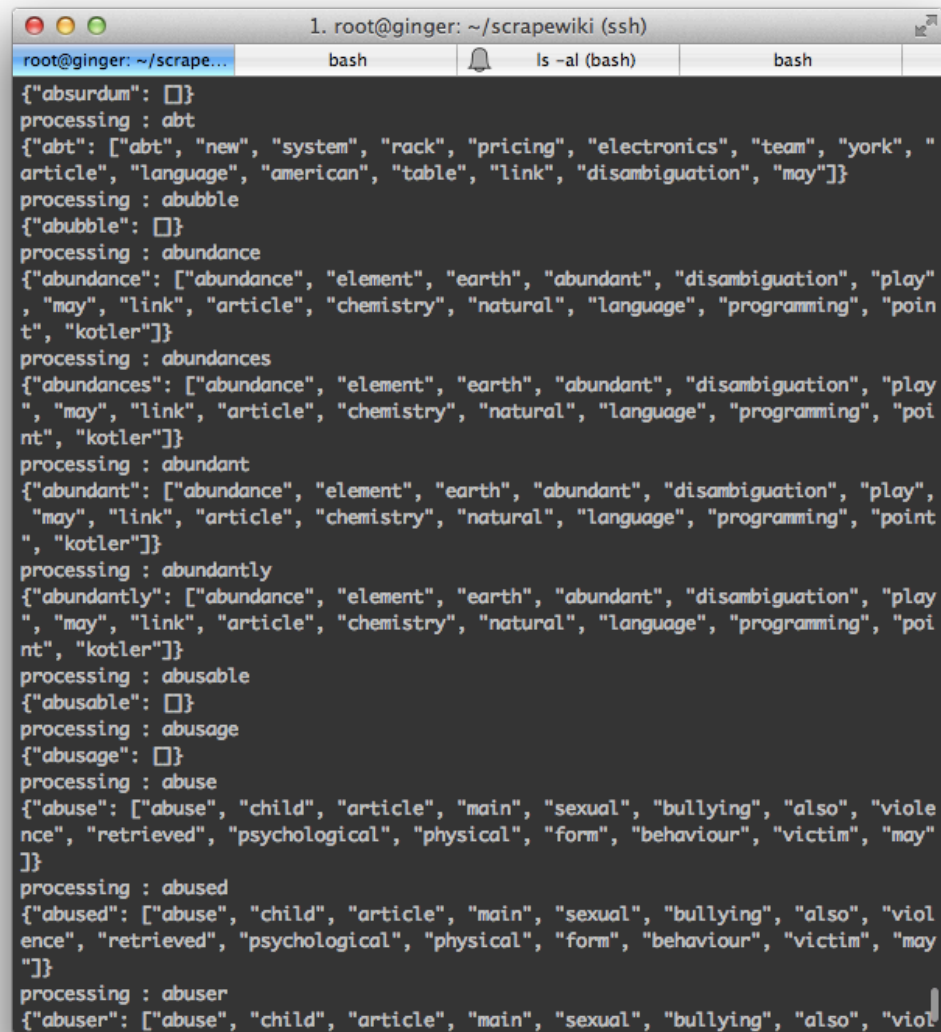
During of the data collection, researcher had to redo three times, because of the tweet data are not suitable for testing as it contained non-English information and unreadable by false Unicode text. Also 30 to 40 percent of the data needed to be clean because contain of hate, discrimination and sexual explicit content. The previous social network data could clean manually because of exhausting curating process.

5.2. Future work and recommendation

During the development, an expert from industry - Mr. Fadhil Luqman, from iEdWrites Management, clarified that the assessment could go more economic if the engine is feed direct data from Twitter.com instead of mining the data itself, as the social network offering Search API and Streamline API to search through their data.

On the scraping and mining side, although the assessment used high performance processing and internet connection (Singapore datacenter, 40Mbps of connection speed and single core of processing power), the rate are still reasonable, approximately ~3.2 to 5.2 seconds to count matrix using Latent Semantic Analysis on a single Wikipedia pages; by download document using Python's urllib2 in UNIX terminal and calculate words frequency of 2000-3400 words of content.

The Synset would be cleaner if excluding unnecessary data for synonym like dates, and common Wikipedia words such as 'see', 'may', 'used' and 'disambiguation'. Even Though lemmatization and stemmer in NLTK (<http://nltk.org>) helps to exclude the common keywords from counted, still there are few common words from Wikipedia are being used by the crowd-sourced platform need to be omitted. According to the synonym schema, the number of completely unrelated data is quite high; 17 to 30% of the collected synonyms are unassociated.

A terminal window titled '1. root@ginger: ~/scrapewiki (ssh)' with tabs for 'root@ginger: ~/scrape...', 'bash', 'ls -al (bash)', and 'bash'. The terminal displays a series of JSON-like outputs for different words, each followed by a 'processing' status. The words shown are 'absurdum', 'abt', 'abubble', 'abundance', 'abundances', 'abundant', 'abundantly', 'abusable', 'abusage', 'abuse', 'abused', and 'abuser'. Each output is a list of related terms in quotes, such as ['abundance', 'element', 'earth', 'abundant', 'disambiguation', 'play', 'may', 'link', 'article', 'chemistry', 'natural', 'language', 'programming', 'point', 'kotler'] for 'abundance'.

```
1. root@ginger: ~/scrapewiki (ssh)
root@ginger: ~/scrape... bash ls -al (bash) bash
{"absurdum": []}
processing : abt
{"abt": ["abt", "new", "system", "rack", "pricing", "electronics", "team", "york", "article", "language", "american", "table", "link", "disambiguation", "may"]}
processing : abubble
{"abubble": []}
processing : abundance
{"abundance": ["abundance", "element", "earth", "abundant", "disambiguation", "play", "may", "link", "article", "chemistry", "natural", "language", "programming", "point", "kotler"]}
processing : abundances
{"abundances": ["abundance", "element", "earth", "abundant", "disambiguation", "play", "may", "link", "article", "chemistry", "natural", "language", "programming", "point", "kotler"]}
processing : abundant
{"abundant": ["abundance", "element", "earth", "abundant", "disambiguation", "play", "may", "link", "article", "chemistry", "natural", "language", "programming", "point", "kotler"]}
processing : abundantly
{"abundantly": ["abundance", "element", "earth", "abundant", "disambiguation", "play", "may", "link", "article", "chemistry", "natural", "language", "programming", "point", "kotler"]}
processing : abusable
{"abusable": []}
processing : abusage
{"abusage": []}
processing : abuse
{"abuse": ["abuse", "child", "article", "main", "sexual", "bullying", "also", "violence", "retrieved", "psychological", "physical", "form", "behaviour", "victim", "may"]}
processing : abused
{"abused": ["abuse", "child", "article", "main", "sexual", "bullying", "also", "violence", "retrieved", "psychological", "physical", "form", "behaviour", "victim", "may"]}
processing : abuser
{"abuser": ["abuse", "child", "article", "main", "sexual", "bullying", "also", "violence", "retrieved", "psychological", "physical", "form", "behaviour", "victim", "may"]}
```

Figure 14 : Synset mining and processing

Because the assessment is depending on Wikipedia alone, we could noticed some of the most frequently words are actually not making any sense. Even though the assessment are focusing on completeness, not on relevancy of the result. There still an attempt to minimize the precision threshold in the research. As the bigger the range of the most term frequent, the precision are gone loose and irrelevant.

Take example of the frequent terms of word “wrong”;

```
{"wrong": ["wrong", "law", "oxford", "legal", "civil", "moral", "crime",  
"damage", "principle", "concept", "press", "justice", "criminal", "offence",  
"university"]}
```

Notice words such as ‘University’, ‘justice’, ‘concept’ and ‘press’ are out of the sense at all. According to research results by respondents, 23 percent of the synonyms outcome is not relevant and not associated at all. This could be minimized the irrelevant by crop down the range, such as drill down to top 5 most frequent terms, for instance.

Latent Semantic Analysis is a good method to gathering and producing a set of concepts related to the documents and terms. But depending on single source like Wikipedia content alone might be inaccurate at some point. It should taking in few considerations to combine with another source of dataset and content for more precise result to the result indexing of search engine.

CHAPTER 6

REFERENCES

6. Reference

6.1. Literatures

1. Dania, B. (2000). Children's use of the Yahoo!igans! Web search engine: I. Cognitive, physical, and affective behaviors on fact-based search tasks. Tennessee: Journal of the American Society for Information Science.
2. Nachmias, R. & Gilad, A. (2002). Needle in a hyperstack: Searching for information on the World Wide Web
3. Tel-Aviv: Journal of research on technology in education.
4. White, R.W., Kules, B., Drucker, S.M., and schraefel, m.c. (2006). Supporting Exploratory Search, Introduction to Special Section of Communications of the ACM, Vol. 49, Issue 4, (2006), pp. 36–39.
5. Sankar, S. (March, 2013). Under the Hood: Indexing and ranking in Graph Search. Retrieved from <https://www.facebook.com/notes/facebook-engineering/under-the-hood-indexing-and-ranking-in-graph-search/10151361720763920>
6. Saracevic, T.(2008). Effects of Inconsistent Relevance Judgments on Information Retrieval Test Results: A Historical Perspective. Library Trends 56(4), 763-783. The Johns Hopkins University Press. Retrieved November 1, 2013, from Project MUSE database.
7. Berners-Lee, T., (n.d.). Linked Data. Retrieved from <http://www.w3.org/standards/semanticweb/data>.
8. John, T , "What is Semantic Search and how it works with Google search". Retrieved December , 2013 Available: <http://www.techulator.com/resources/5933-What-Semantic-Search.aspx>
9. Biesnecker, J , "What are the top startup engineering blogs?". Retrieved December , 2013 Available: <http://www.quora.com/What-are-the-top->

startup-engineering-blogs.

10. Reis, E. (2011). The Lean Startup. Crown Business.
<http://www.amazon.com/The-Lean-Startup-Entrepreneurs-Continuous/dp/0307887898>
11. Karypis, G., Han, E., (2000). Fast Supervised Dimensionality Reduction Algorithm with Applications to Document Categorization and Retrieval, Proceedings of CIKM-00, 9th ACM Conference on Information and Knowledge Management.
12. Gabrilovich, E. & Markovitch, S. (2006). Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis.

6.2. Datasets

1. English Wordlist, SIL International Linguistics Department. (2005).
“Retrieved from <http://www-01.sil.org/linguistics/wordlists/english>“
2. English Wikipedia, Wikimedia Foundation. (2013). “Retrieved from http://dumps.wikimedia.org/other/static_html_dumps/current/en/wikipedia-en-html.tar.7z”

CHAPTER 7

APPENDIX

7 Appendix

7.1 Data validation survey

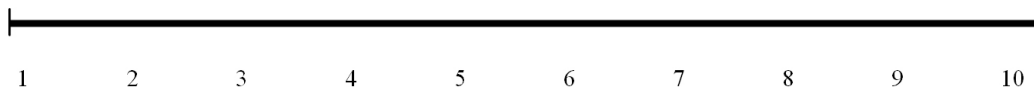
Set 1

This survey is intended to get feedback of relevancy of the data set for semantic search assessment titled; Search Result Synonymy Indexing for Social Network Using Latent Semantic Analysis. There are 10 data of synonym data known as synonym schema.

This study is using empirical method meant to get feedback to collect score represent of the whole clause. Respondent may scale the data from 1 to 10 for each.

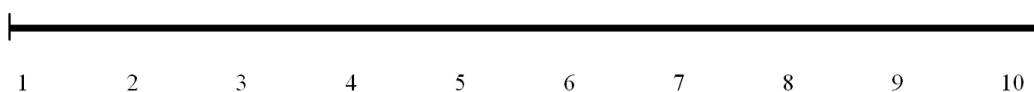
Data 1

{"wok": ["wok", "stove", "cooking", "chinese", "food", "heat", "used", "steel", "iron", "cast", "also", "frying", "traditional", "handle", "stir"]}



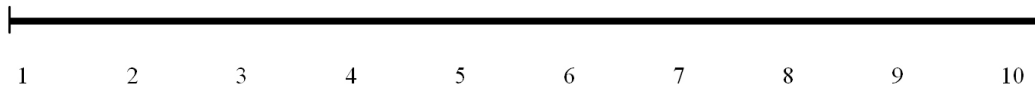
Data 2

{"Software": ["computer", "system", "application", "user", "language", "program", "web", "programming", "microsoft", "may", "tool", "instruction", "embedded", "use", "one"]}



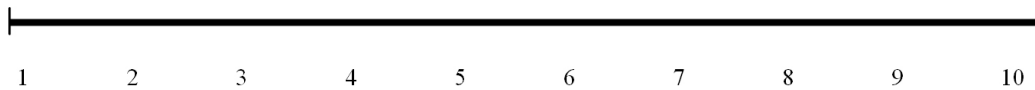
Data 3

{"wok": ["wok", "stove", "cooking", "chinese", "food", "heat", "used", "steel", "iron", "cast", "also", "frying", "traditional", "handle", "stir"]}



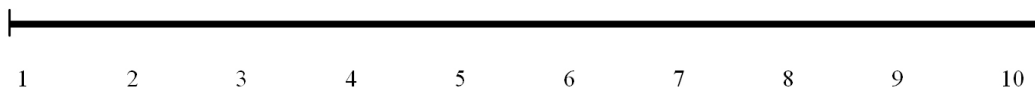
Data 4

{"Warning": ["warning", "film", "album", "song", "band", "black", "sabbath", "see", "article", "2000", "1946", "stop", "traffic", "day", "signal"]}



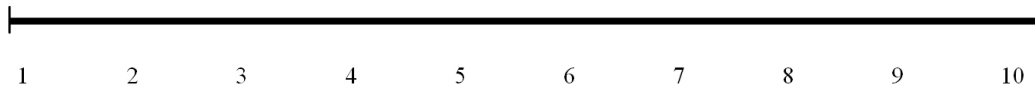
Data 5

{"Create": ["wok", "stove", "cooking", "chinese", "food", "heat", "used", "steel", "iron", "cast", "also", "frying", "traditional", "handle", "stir"]}



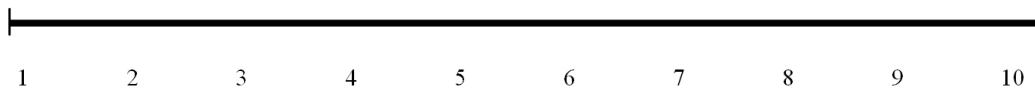
Data 6

{ "Journal": ["journal", "public", "business", "record", "also", "magazine", "transaction", "daily", "writing", "day", "see", "event", "called", "scholarly", "reference"] }



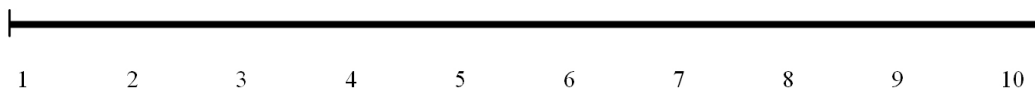
Data 7

{ "aardvark": ["aardvark", "2013", "burrow", "termite", "orycteropus", "animal", "afer", "ant", "dig", "africa", "one", "also", "african", "specie", "long"] }



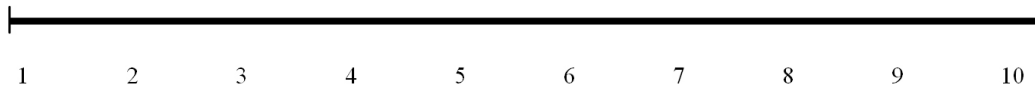
Data 8

{ "abalone": ["haliotis", "abalone", "synonym", "1846", "reeve", "shell", "1758", "linnaeus", "specie", "2010", "tuberculata", "world", "new", "marine", "1822"] }



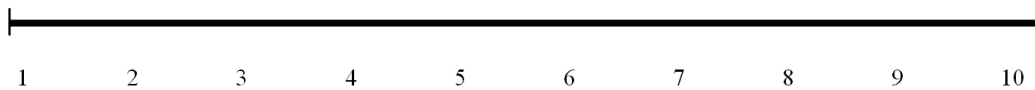
Data 9

```
{"aardvark": ["aardvark", "2013", "burrow", "termite", "orycteropus", "animal", "afer", "ant",  
"dig", "africa", "one", "also", "african", "specie", "long"]}
```



Data 10

```
{"wok": ["wok", "stove", "cooking", "chinese", "food", "heat", "used", "steel", "iron", "cast",  
"also", "frying", "traditional", "handle", "stir"]}
```



Thank you.

Prepared by : i@firdaus.my

7.2 Source codes

7.2.1 Wikipedia data miner and scraper

```
# -*- coding: utf-8 -*-
import urllib2
import codecs
from scrapy.selector import Selector
from nltk.stem.lancaster import LancasterStemmer
from nltk.corpus import stopwords
import re
import nltk
from nltk.tokenize import RegexpTokenizer
from nltk import bigrams, trigrams
import math
import json
from nltk.stem import WordNetLemmatizer
stopwords = nltk.corpus.stopwords.words('english')
tokenizer = RegexpTokenizer("[\w']+", flags=re.UNICODE)
#st = LancasterStemmer()
wnl = WordNetLemmatizer()
keywords=[]
with open('keywords.txt','r') as f:
    for i in f:
        keywords.append(i.strip())

with open('stopwords.txt','r') as f:
    for i in f:
        stopwords.append(i.strip())
def freq(word, doc):
    return doc.count(word)

def word_count(doc):
    return len(doc)

def tf(word, doc):
    return (freq(word, doc) / float(word_count(doc)))

def calcu_tf(keyword):
    url = "http://en.wikipedia.com/wiki/"+keyword
    content = urllib2.urlopen(url).read()

    vocabulary = []

    all_tips = []
    sel=Selector(text=content)
```

```

#text="".join(sel.xpath('//div[@id="mw-content-text"]/*[self::p or
self::ul]//text()).extract())
text="".join(sel.xpath('//div[@id="mw-content-text"]//text()).extract())
if text.find(keyword+' may refer to:') >=0 :
    #if text.find(keyword+' may refer to:') or text.find('Wikipedia does not have an
article with this exact name') >=0 :
    #if text.find('Wikipedia does not have an article with this exact name') >=0 :
        return #[]
tokens = tokenizer.tokenize(text)

tokens = [token.lower() for token in tokens if len(token) > 2]
tokens = [wnl.lemmatize(token) for token in tokens if token not in stopwords]

docs = { 'tf': {}, 'tokens': []}
for token in tokens:

    docs['tf'][token] = tf(token, tokens)

tops=sorted(docs['tf'].items(), key=lambda x: x[1], reverse=True)[:15]
#print tops
return [ i[0] for i in tops ]

with codecs.open('links.txt','ab+', 'utf-8') as f1:
    with codecs.open('tf.json','ab+', 'utf-8') as f:
        j=0
        for i in f :
            print i
            j+=1
        print j
        for keyword in keywords[j:] :
            print "processing : %s "% keyword
            list=[]
            try:
                list=calcu_tf(keyword)
            except urllib2.HTTPError, err:
                if err.code == 404:
                    pass
                else:
                    raise
            f.write(json.dumps({keyword:list})+'\n')
            f1.write("http://en.wikipedia.com/wiki/"+keyword+"\n")
            print json.dumps({keyword:list})
            #break

raw_input()

```


7.2.2 Tweets collector

```
"""
Twitter Stream To MongoDB (c) by gdelfresno

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program. If not, see <http://www.gnu.org/licenses/>.
"""

import threading
import datetime
import time
import re

from optparse import OptionParser
from pymongo import Connection

from ssl import SSLError

from tweepy.streaming import StreamListener
from tweepy import OAuthHandler, BasicAuthHandler
from tweepy import Stream, API
from tweepy.utils import import_simplejson

json = import_simplejson()

active_terms = {}

STREAM_URL = "https://stream.twitter.com/1/statuses/filter.json?track=%s"
streamThread = None

def get_parser():
    parser = OptionParser()
    parser.add_option("-d", "--database", dest="database", help="mongodb database name")
    parser.add_option("-s", "--server", dest="server", help="mongodb host")
    parser.add_option("-p", "--port", dest="port", help="mongodb port", type="int")
    parser.add_option("-a", "--dbauth", dest="dbauth", help="db auth file")
    parser.add_option("-o", "--oauth", dest="oauthfile", help="file with oauth options")
```

```

    parser.add_option("-t", "--track", dest="track", help="track terms file")
    #parser.add_option("-a", "--all_keys", dest="all_keys", help="MongoDB receives
all keys from json (True/False)")
    #parser.add_option("-u", "--use_keys", dest="use_keys", help="MongoDB
receives subset of keys from json (list)")
    parser.add_option("-e", "--exclude-retweets", action="store_true",
dest="exclude_retweets", help="Exclude retweets from stream", default=False)
    parser.usage = "bad parametres"
    return parser

def updateSearchQuery(options):
    print 'Updating terms: %s' % datetime.datetime.now()

    query = ",".join(active_terms.keys())

    try:
        streamThread.stopConsume()

    except:
        pass

    streamThread = StreamConsumerThreadClass(query, options.oauthfile)
    streamThread.setDaemon(True)

    streamThread.start()

def addTerm(term):
    if term == "":
        return

    active_terms[term] = term

def deleteTerm(term):
    if term == "":
        return

    active_terms.pop(term)

def updateTerms(options):
    fileterms = []

    update = False
    f = file(options.track, 'r')
    for line in f.readlines():
        term = str.strip(line)

```

```

        fileterms.append(term)
        #Check new terms
        if not term in active_terms.keys():
            print "New Term: %s" % term
            addTerm(term)
            update = True

    for current in active_terms.keys():
        if not current in fileterms:
            print "Deleted term: %s" % current
            deleteTerm(current)
            update = True

    if update:
        updateSearchQuery(options)

def prettyPrintStatus(status):
    text = status["text"]
    description = status['user']['screen_name']
    if "retweeted_status" in status:
        description = ("%s RT by %s" %
(status["retweeted_status"]["user"]["screen_name"], status['user']['screen_name']))
        text = status["retweeted_status"]["text"]

    try:
        return "[%s][%-36s]: %s" % (status['created_at'], description, text)
    except UnicodeEncodeError:
        return "Can't decode UNICODE tweet"
    except :
        return "Error printing status"

class MongoDBCoordinator:
    def __init__(self, host='localhost', port=None, database='TwitterStream',
authfile=None):
        try:
            if not port is None:
                self.mongo = Connection(host, int(port))
            else:
                self.mongo = Connection(host)
        except:
            print "Error starting MongoDB"
            raise

        self.db = self.mongo[database]

        if not authfile is None:

```

```

    try:
        dbauth = json.loads(open(authfile, 'r').read())
        if not self.db.authenticate(dbauth["user"], dbauth["password"]):
            raise Exception("Invalid database credentials")

    except:
        print "Error authenticating database"
        raise

self.tuits = {}

def addTuit(self, tweet):
    for term in active_terms.keys():

        if "retweeted_status" in tweet:
            content = tweet["retweeted_status"]["text"]
        else:
            content = tweet['text']

        stre = re.compile(term, re.IGNORECASE)
        match = stre.search(content)

        if match:
            if not term in self.db.collection_names():
                self.db.create_collection(term)

            collection = self.db[term]
            collection.save(tweet)

            try:
                print "[%15s] %s" % (term, prettyPrintStatus(tweet))
            except Exception as (e):
                print "Error %s" % e.message

class MongoDBListener(StreamListener):
    """
    A listener handles tweets are the received from the stream.
    This is a basic listener that just prints received tweets to stdout.
    """
    def on_data(self, data):
        """
        Called when raw data is received from connection.

        Override this method if you wish to manually handle
        the stream data. Return False to stop stream and close connection.
        """
        if 'retweeted_status' in data:
            if options.exclude_retweets:

```

```

        pass
    else:
        jstatus = json.loads(data)
        mongo.addTuit(jstatus)

    elif 'in_reply_to_status_id' in data:
        jstatus = json.loads(data)
        mongo.addTuit(jstatus)

    elif 'delete' in data:
        delete = json.loads(data)['delete']['status']
        if self.on_delete(delete['id'], delete['user_id']) is False:
            return False

    elif 'limit' in data:
        if self.on_limit(json.loads(data)['limit']['track']) is False:
            return False

    def on_error(self, status_code):
        if status_code == 401:
            raise Exception("Invalid logging credentials")
        else:
            print "Error received %d" % status_code

    def on_limit(self, track):
        print "##### LIMIT ERROR #####"

#Class that track the stream
class StreamConsumerThreadClass(threading.Thread):
    def __init__(self, term="", oauthfile=""):
        threading.Thread.__init__(self)
        self.searchterm = term
        self.name = term
        self.consume = True

    listener = MongoDBListener()

    try:
        oauth = json.loads(open(oauthfile, 'r').read())

        if 'consumer_key' in oauth:
            auth = OAuthHandler(oauth['consumer_key'], oauth['consumer_secret'])
            auth.set_access_token(oauth['access_token'], oauth['access_token_secret'])
            api = API(auth)

            if not api.verify_credentials():
                raise Exception("Invalid credentials")
            else:

```

```

        auth = BasicAuthHandler(oauth['username'], oauth['password'])

    except:
        print "Error logging to Twitter"
        raise

    self.stream = Stream(auth, listener, timeout=60)

def stopConsume(self):
    self.stream.disconnect()

def run(self):
    now = datetime.datetime.now()
    print "Twitter Stream with terms: %s started at: %s" % (self.getName(), now)

    connected = False
    while True:
        try:
            if not connected:
                connected = True
                self.stream.filter(track=[self.searchterm])

        except SSLError, e:
            print e
            connected = False
        except Exception, e:
            print "Stream error"
            raise e

if __name__ == "__main__":
    parser = get_parser()
    (options, args) = parser.parse_args()
    print options, args

    try:
        mongo = MongoDBCoordinator(options.server, options.port, options.database,
options.dbauth)
        streamThread = StreamConsumerThreadClass("", options.oauthfile)
    except Exception, e:
        print e
        exit(0)

    try:
        while True:
            updateTerms(options)
            time.sleep(5)

    except KeyboardInterrupt, e:

```

```
print "Closing stream"  
streamThread.stopConsume()
```